Modelling of Software intensive systems

Assignment 2 CBD

First Edition

L. Jason	Jason.Liu@student.uantwerpen.be	s0213082
S. Kin Ning	KinNing.Shum@student.uantwerpen.be	s0202054

2024 - 12 - 21

Contents

1 | Integration Methods

In this section, we numerically approximate integration using multiple integration methods. These methods are implemented with Causal Block Diagrams.

1.1 | Backwards Euler

Formula:

$I_0 = f(t_0) \cdot \Delta t$
$I_1 = f(t_1) \cdot \Delta t$
$I = I_0 + I_1$

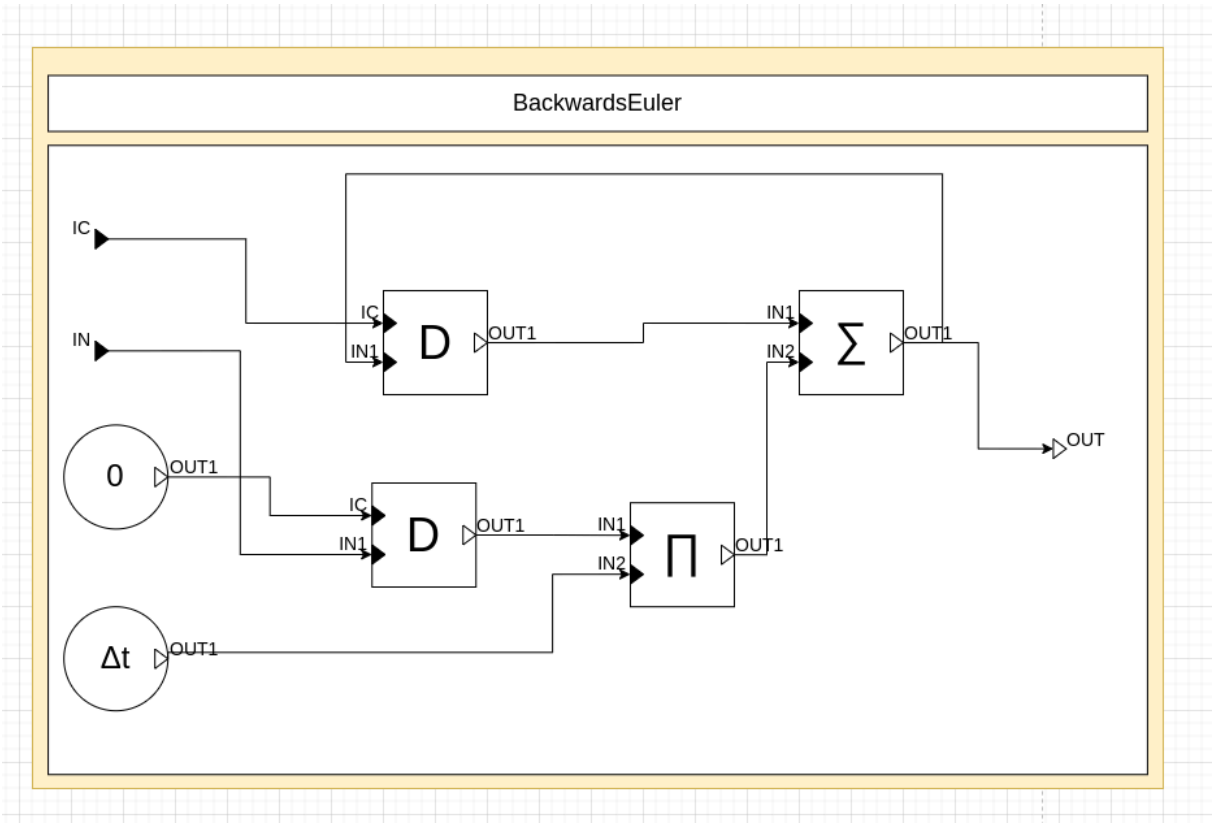

Figure 1: Backwards Euler diagram
assets/part_1/Backwards_Euler_drawio.png

This is an exact copy of the Backwards Euler implementation in pyCBD. It multiplies the input with Δt and adds it to the output of the previous iteration. The first delay block outputs the result of the previous iteration. The second one ensures the first iteration will always output the value of IC, by multiplying 0 with Δt and adding it to IC.

1.2 | Forwards Euler

Formula:

$I_0 = f(t_1) \cdot \Delta t$
$I_1 = f(t_2) \cdot \Delta t$
$I = I_0 + I_1$

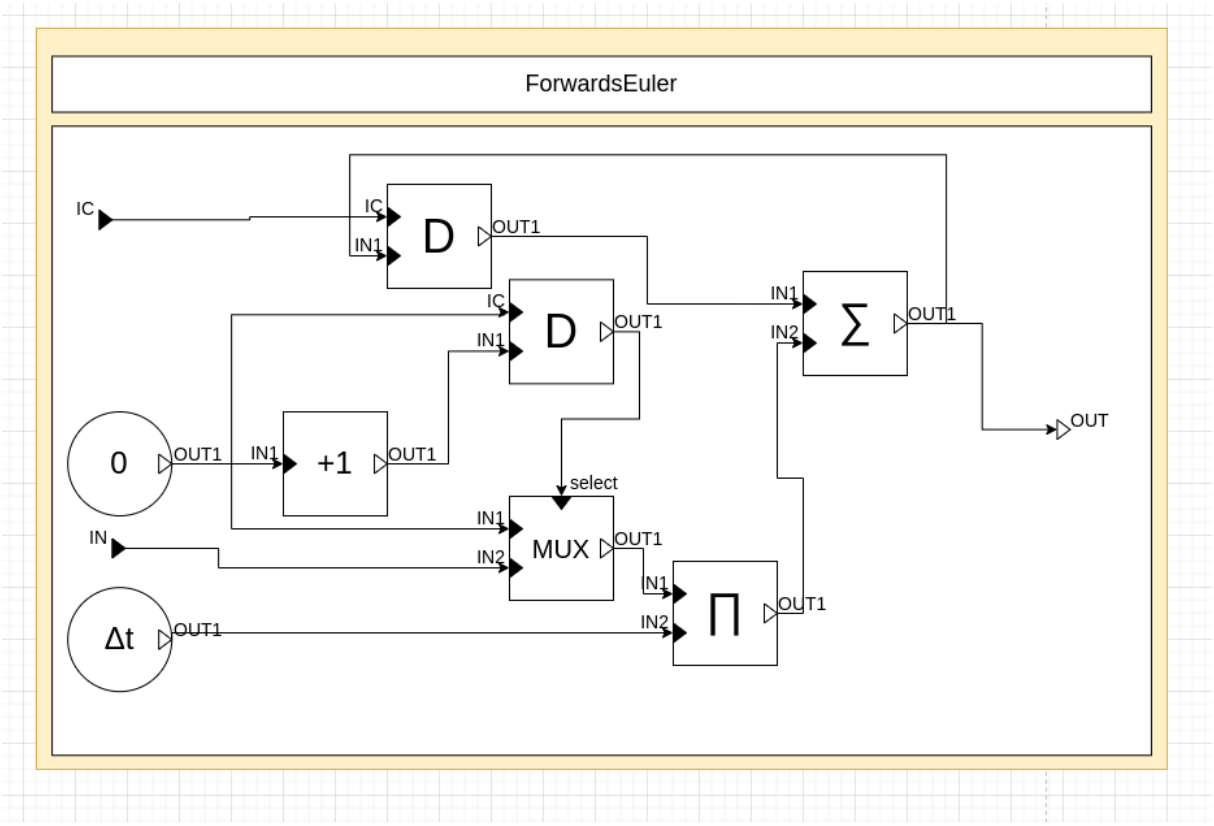

Figure 2: Forwards Euler diagram
assets/part_1/Forwards_Euler_drawio.png

While it looks very different to the Backwards Euler implementation, it is actually very similar. The second delay block is replaced by a multiplexer, and some logic is added for the multiplexer selection. The value of $f(t_1)$ is used instead of $f(t_0)$, so we don't need a delay block. However, we need to ensure IC is the output in the first iteration, so we need to replace $f(t_0)$ with 0. To do that, a multiplexer and a delay block is used. The delay block outputs 0 in the first iteration and 1 otherwise. The rest of the logic is exactly the same as in the Backwards Euler implementation.

As the input $f(t)$ is not fed into a delay block, Forwards Euler will use the value of $f(t_{i+1})$, while Backwards Euler will use $f(t_i)$ in the same iteration, just like the formula. As of writing this, I have realized I could have used a constant block instead of an add one block.

1.3 | Trapezoid Rule

Formula:

$I_0 = \frac{f(t_0)+f(t_1)}{2} \cdot \Delta t$
$I_1 = \frac{f(t_1)+f(t_2)}{2} \cdot \Delta t$
$I = I_0 + I_1$

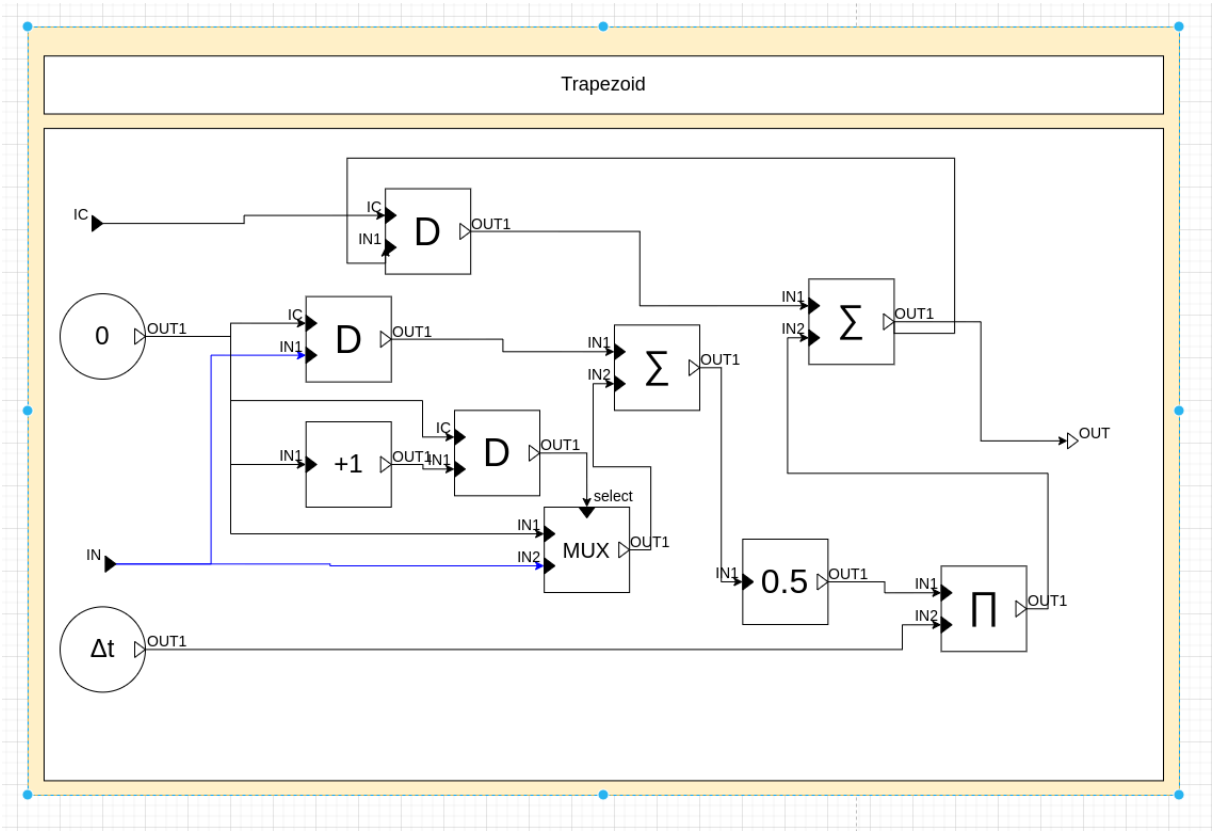

Figure 3: Trapezoid Rule diagram, some arrows have different colors for clarity purposes. assets/part_1/Trapezoid_drawio.png

Now we need both t_0 and t_1. We get them using the same method as in the previous diagrams and combine them to get them both in the same iteration. Now we just add them up and divide by 2. The rest of the logic is the same as the previous two methods. Just like in Forwards Euler, a constant block could have been used instead of an add one block.

1.4 | g(t)

The function is represented as the following:

$g(t) = \frac{t+2}{(t+3)^2}$

The associated diagram is straightforward and does not need any additional explanations. However, one thing to note is that we multiplied $(t + 3)$ with itself instead of raising it to the power of 2 to achieve the same results.

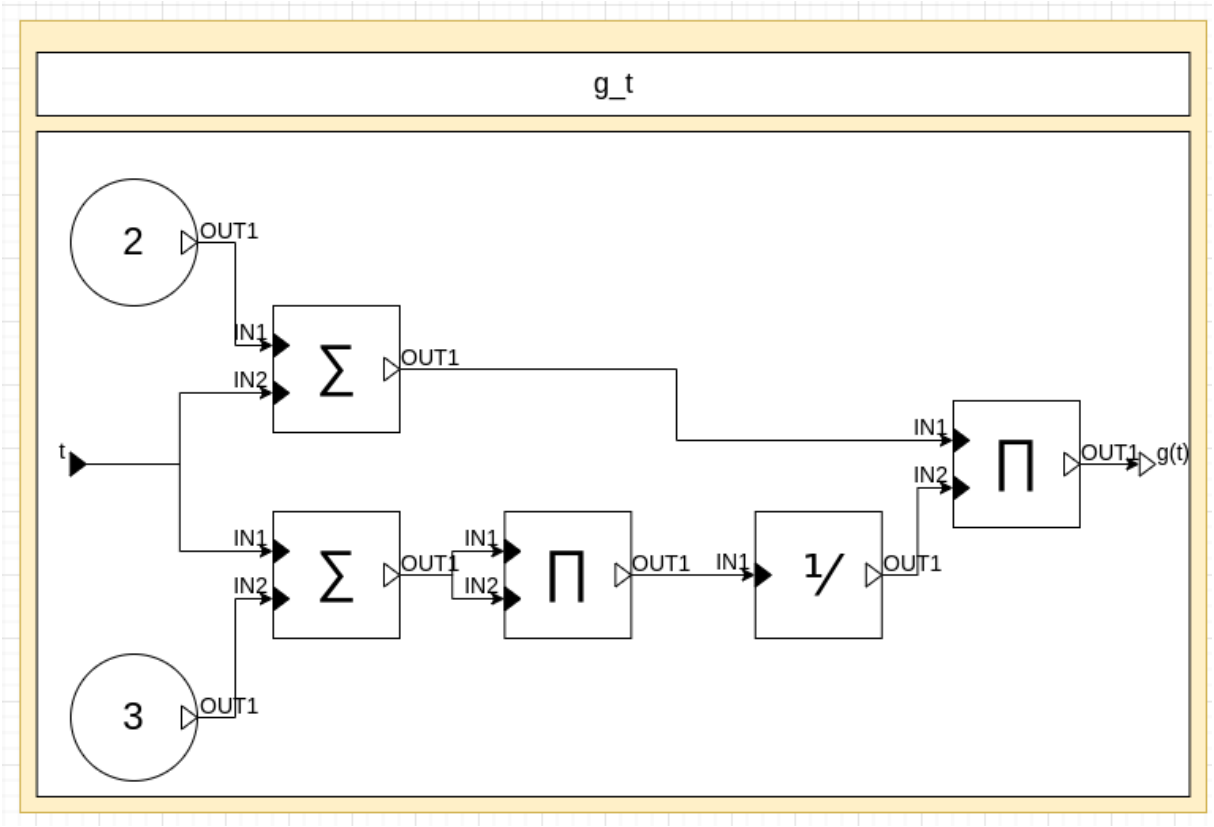

Figure 4: g(t) diagram
assets/part_1/g_t_drawio.png

1.5 | Comparison

We compare the integration methods with each other and analyze it numerically. We do so by calculating $\int_0^{100} G(t)dt$ with each method and with different values of Δt.

The analytical value using Wolfram Alpha: 3.21249210409227…

Δt	Backwards Euler	Forwards Euler	Trapezoid Rule
0.1	3.222190908877023	3.2009310596645353	3.2115609842707813
0.01	3.213459296657502	3.2113332284618306	3.2123962625596754
0.001	3.212588796472303	3.212376188820784	3.2124824926465383

Figure 5: Integration simulation results table
assets/part_1/integration_table.jpeg

We notice that Backwards Euler always overshoots, Forwards Euler always undershoots, and the Trapezoid Rule also undershoots, but less than the Forward Euler. This can be explained if we look at the graph of $g(t)$.

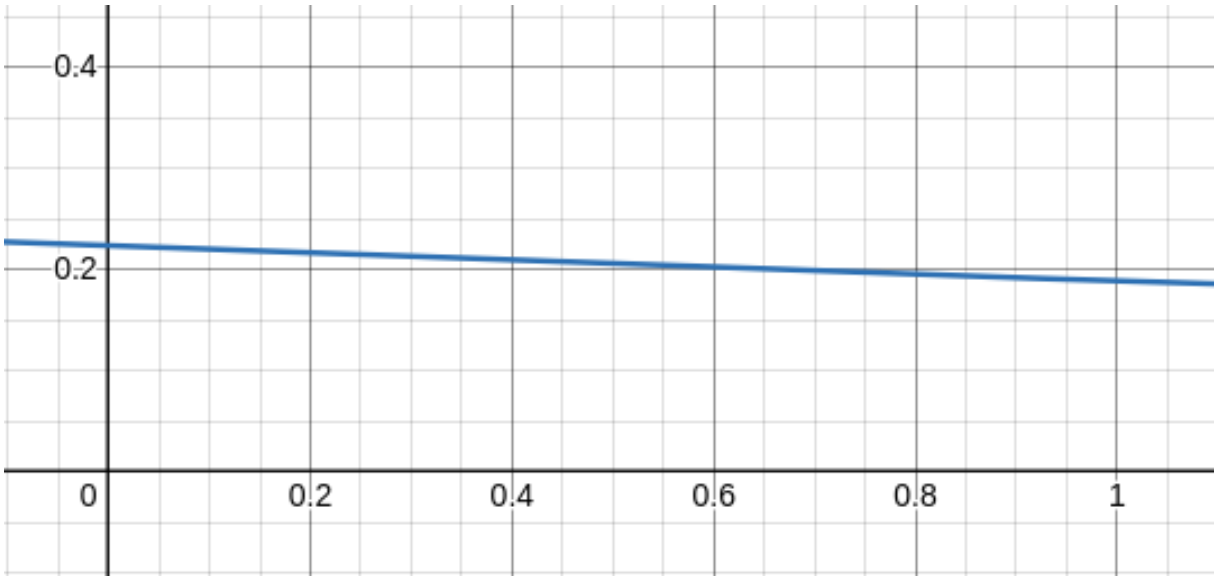

Figure 6: Graph of $g(t)$
assets/part_1/g_t_graph.png

It is clearly a downwards slope. The Backwards Euler approximates by using a smaller t, which leads to it overshooting the actual value, as $g(t)$ is a downwards slope for $t \geq 0$. The exact opposite happens with Forward Euler, leading to it overshooting. If $g(t)$ was an upwards slope, however, then Backwards Euler would undershoot and Forwards Euler would overshoot.

As for the Trapezoid Rule, if we look at the formula, we can see that it is actually just the mean of the two Euler methods. The difference between them in a simulation with $\Delta t = 0.001$ is 0.00000000000000532907. The numerical precision of 64 bit floating point is about 15 digits, so the difference is very likely due to numerical errors.

Now let's look at the error of the integration approximations compared to the analytical result.

Δt	Backwards Euler Error	Forwards Euler Error	Trapezoid Rule Error
0.1	0.09698804784752823 59	0.01156104442773475682	0.00093111982148874617
0.01	0.00096719256523192243	0.00115887563043948205	0.00009584153259467598
0.001	0.00009669238003295888	0.00011591527148624436	0.00000961144573174977

Figure 7: Integration simulation results error table
assets/part_1/integration_error.jpeg

Higher precision leads to lower errors, which is obvious. Forwards Euler has the highest error, although Backwards Euler has a similar error. Trapezoid Rule has much smaller error, around 10 times lower.

Considering the accuracy, using the Trapezoid Rule may be the optimal choice. However, if we consider the computation time, that may not be the case. These are the average computation times of Backwards Euler, Forwards Euler and Trapezoid Rule respectively: 2.5 seconds, 3 seconds, 3.5 seconds. Notice that Trapezoid Rule took 40% longer than

Backwards Euler. Forwards Euler also took longer, while having similar precision. Note that our implementation of Forwards Euler and Trapezoid Rule is not optimal. However, the results will not be much different even if our implementations are optimal.

We conclude that Backwards Euler is the most efficient method out of the three, sacrificing a little bit of accuracy. If precision is most important, Trapezoid Rule is the best, with higher computation.

2 | Co-simulation

We start by exporting the FMU model of the plant, and we will make use of this model to co-simulate with our CBD model.

2.1 | CBD model making use of Drawio

We will make use of Drawio to draw our CBD model. It will look something like this:

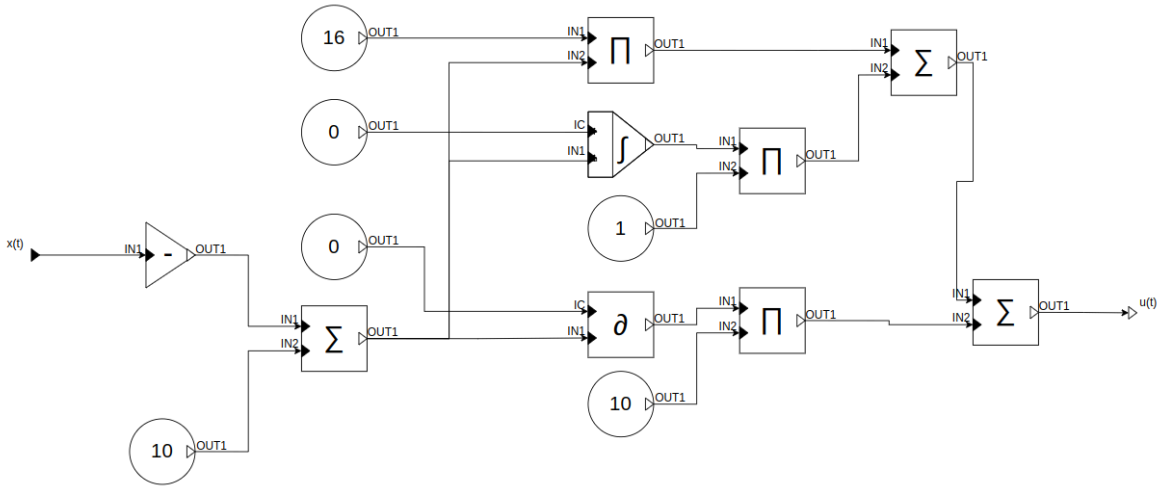

Figure 8: PID circuit
assets/part_2/PID.png

Here we will reuse the values we got from the previous assignment and make use of $K_i = 1$ instead of 0. Notice how we use Product Blocks and Constant Blocks to multiply with a constant. We realized later that we could have used Gain Blocks to make it simpler, but we have already exported the diagrams to FMU. As it actually does not matter that much, we spared the effort and left it as is.

We can turn this circuit also in a block and this block will look as follows:

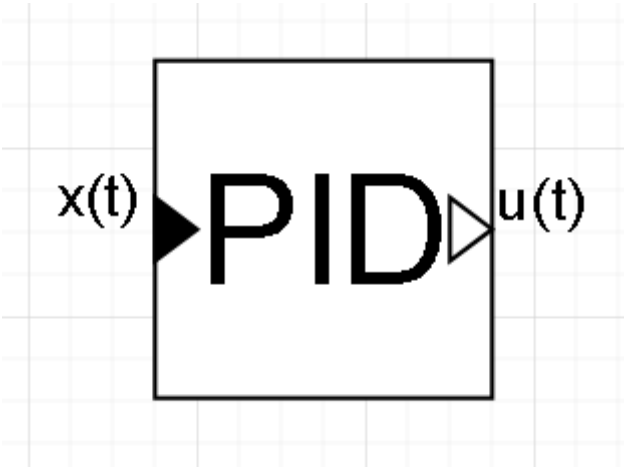

Figure 9: PID block
assets/part_2/PID_block.png

With a simple input of x(t) and simple output of the control signal u(t).

The full drawio schema makes use of a class diagram, this will look something like this:

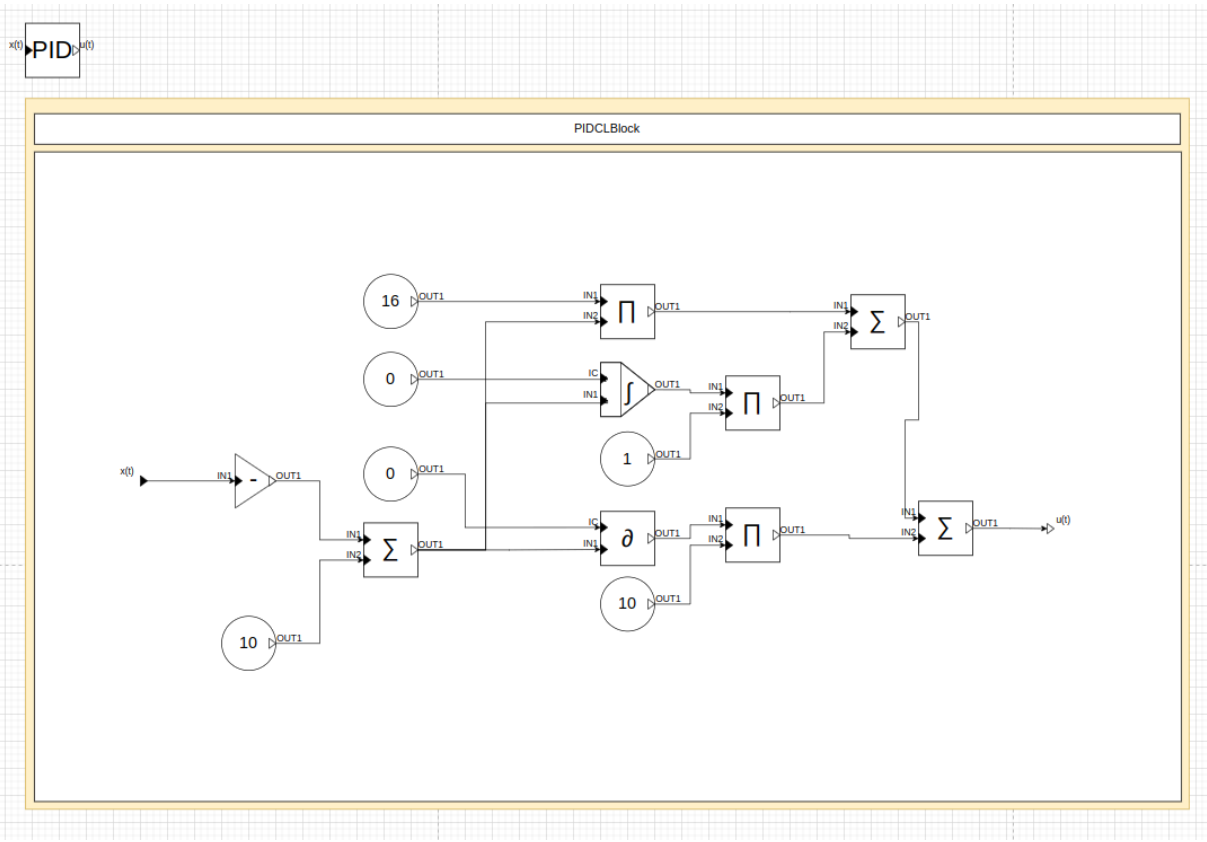

Figure 10: PID full schema
assets/part_2/PID_full.png

We will call this block PIDCLBLock. This is important to know for later.

2.2 | Controller FMU from Controller CBD

The description.xml will have the following contents:

```
<?xml version="1.0" encoding="UTF-8"?>
<fmiModelDescription fmiVersion="2.0" modelName="PIDCLBlock"
  guid="{1c15c3ee-a843-4c31-b39d-9fe699fc1e63}"
  description="" generationTool="CBD2FMU">

  <ModelExchange modelIdentifier="PIDCLBlock">
    <SourceFiles>
      <File name="all.c"/>
      <File name="lsolve.c"/>
    </SourceFiles>
  </ModelExchange>
  <CoSimulation modelIdentifier="PIDCLBlock"
canHandleVariableCommunicationStepSize="true">
    <SourceFiles>
      <File name="all.c"/>
      <File name="lsolve.c"/>
    </SourceFiles>
  </CoSimulation>
```

```
  <!-- Unit definitions: Not required/used in CBD -->
  <!-- Log categories: Not required/used in CBD -->

  <DefaultExperiment startTime="0.0"
    stopTime="20"
    stepSize="0.004"
    tolerance="1e-06" />

  <!-- Vendor Annotations: Not required/used in CBD -->

  <ModelVariables>
    <ScalarVariable name="PIDCLBlock.x(t)" valueReference="0"
      causality="input" variability="continuous">
      <Real start="0"/>
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.u(t)" valueReference="1" initial="calculated"
      causality="output" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-4.IN1" valueReference="2"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-4.OUT1" valueReference="3"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-7.IN1" valueReference="4"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-7.IN2" valueReference="5"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-7.OUT1" valueReference="6"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-11.OUT1" valueReference="7"
initial="exact"
      causality="local" variability="constant">
      <Real start="10"/>
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-21.IN1" valueReference="8"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-21.IN2" valueReference="9"
initial="calculated"
```

```
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-21.OUT1"
valueReference="10" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-25.IN1" valueReference="11"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-25.IN2" valueReference="12"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-25.OUT1"
valueReference="13" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-29.IN1" valueReference="14"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-29.IN2" valueReference="15"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-29.OUT1"
valueReference="16" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.K_p.OUT1" valueReference="17" initial="exact"
      causality="local" variability="constant">
      <Real start="16"/>
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.IC_d.OUT1" valueReference="18" initial="exact"
      causality="local" variability="constant">
      <Real start="0"/>
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.IC_i.OUT1" valueReference="19" initial="exact"
      causality="local" variability="constant">
      <Real start="0"/>
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.dqvIs03bip9LbBkwujtJ-2.IN1" valueReference="20"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.dqvIs03bip9LbBkwujtJ-2.IN2" valueReference="21"
```

```
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.dqvIs03bip9LbBkwujtJ-2.OUT1" valueReference="22"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.K_i.OUT1" valueReference="23" initial="exact"
      causality="local" variability="constant">
      <Real start="1"/>
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.K_d.OUT1" valueReference="24" initial="exact"
      causality="local" variability="constant">
      <Real start="10"/>
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.dqvIs03bip9LbBkwujtJ-76.IN1" valueReference="25"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.dqvIs03bip9LbBkwujtJ-76.IN2" valueReference="26"
initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.dqvIs03bip9LbBkwujtJ-76.OUT1"
valueReference="27" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_zero.OUT1"
valueReference="28" initial="exact"
      causality="local" variability="constant">
      <Real start="0"/>
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_delta_t.OUT1"
valueReference="29" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_delayIn.IN1"
valueReference="30" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_delayIn.IC"
valueReference="31" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_delayIn.OUT1"
valueReference="32" initial="calculated"
      causality="local" variability="continuous">
      <Real />
```

```
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_multDelta.IN1"
valueReference="33" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_multDelta.IN2"
valueReference="34" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_multDelta.OUT1"
valueReference="35" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_delayState.IN1"
valueReference="36" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_delayState.IC"
valueReference="37" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_delayState.OUT1"
valueReference="38" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_sumState.IN1"
valueReference="39" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_sumState.IN2"
valueReference="40" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-13_sumState.OUT1"
valueReference="41" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_delta_t.OUT1"
valueReference="42" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_multIc.IN1"
valueReference="43" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
```

```
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_multIc.IN2"
valueReference="44" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_multIc.OUT1"
valueReference="45" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_neg1.IN1"
valueReference="46" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_neg1.OUT1"
valueReference="47" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_sum1.IN1"
valueReference="48" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_sum1.IN2"
valueReference="49" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_sum1.OUT1"
valueReference="50" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_delay.IN1"
valueReference="51" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_delay.IC"
valueReference="52" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_delay.OUT1"
valueReference="53" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_neg2.IN1"
valueReference="54" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_neg2.OUT1"
```

```
valueReference="55" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_sum2.IN1"
valueReference="56" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_sum2.IN2"
valueReference="57" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_sum2.OUT1"
valueReference="58" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_mult.IN1"
valueReference="59" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_mult.IN2"
valueReference="60" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_mult.OUT1"
valueReference="61" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_inv.IN1"
valueReference="62" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
    <ScalarVariable name="PIDCLBlock.DzCg6ToobE1hvCms1og_-17_inv.OUT1"
valueReference="63" initial="calculated"
      causality="local" variability="continuous">
      <Real />
    </ScalarVariable>
  </ModelVariables>

  <ModelStructure>
    <Outputs>
      <Unknown index="2"/>
    </Outputs>
    <InitialUnknowns>
      <Unknown index="2"/>
    </InitialUnknowns>
  </ModelStructure>
</fmiModelDescription>
```

The C-code will look something like this:

```c
#include "lsolve.h"
#include "version.h"
#include "model.h"
#include <stdio.h>
#include <math.h>

void initialEquations(CBD* cbd) {
  Real delta = 0.0001;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__4_IN1 = _PIDCLBlock_x(t);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__4_OUT1 = (-
_PIDCLBlock_DzCg6ToobE1hvCms1og__4_IN1);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__11_OUT1 = 10;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__7_IN2 = _PIDCLBlock_DzCg6ToobE1hvCms1og__11_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__7_IN1 = _PIDCLBlock_DzCg6ToobE1hvCms1og__4_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1 = (_PIDCLBlock_DzCg6ToobE1hvCms1og__7_IN1 +
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_IN2);
  _PIDCLBlock_K_p_OUT1 = 16;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__29_IN2 = _PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__29_IN1 = _PIDCLBlock_K_p_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__29_OUT1 = (_PIDCLBlock_DzCg6ToobE1hvCms1og__29_IN1
* _PIDCLBlock_DzCg6ToobE1hvCms1og__29_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_zero_OUT1 = 0;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_IC =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_zero_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_OUT1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_IC;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delta_t_OUT1 = delta;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delta_t_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_IN1 *
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_IN2);
  _PIDCLBlock_IC_i_OUT1 = 0;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_IC = _PIDCLBlock_IC_i_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_OUT1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_IC;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_IN1 +
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_IN2);
  _PIDCLBlock_K_i_OUT1 = 1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_IN2 = _PIDCLBlock_K_i_OUT1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_OUT1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_OUT1 = (_PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_IN1 *
_PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__21_IN2 = _PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_OUT1;
```

```
  _PIDCLBlock_DzCg6ToobE1hvCms1og__21_IN1 = _PIDCLBlock_DzCg6ToobE1hvCms1og__29_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__21_OUT1 = (_PIDCLBlock_DzCg6ToobE1hvCms1og__21_IN1
+ _PIDCLBlock_DzCg6ToobE1hvCms1og__21_IN2);
  _PIDCLBlock_IC_d_OUT1 = 0;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_delta_t_OUT1 = delta;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_delta_t_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_IN1 = _PIDCLBlock_IC_d_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_IN1 *
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg1_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg1_OUT1 = (-
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg1_IN1);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg1_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_IN1 +
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_IC =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_OUT1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_IC;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg2_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg2_OUT1 = (-
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg2_IN1);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg2_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_IN1 +
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_inv_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_delta_t_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_inv_OUT1 = 1.0/
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_inv_IN1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_inv_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_IN1 *
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_IN2);
  _PIDCLBlock_K_d_OUT1 = 10;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_IN2 = _PIDCLBlock_K_d_OUT1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_OUT1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_OUT1 = (_PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_IN1
* _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_IN2);
```

```
  _PIDCLBlock_DzCg6ToobE1hvCms1og__25_IN2 = _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__25_IN1 = _PIDCLBlock_DzCg6ToobE1hvCms1og__21_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__25_OUT1 = (_PIDCLBlock_DzCg6ToobE1hvCms1og__25_IN1
+ _PIDCLBlock_DzCg6ToobE1hvCms1og__25_IN2);
  _PIDCLBlock_u(t) = _PIDCLBlock_DzCg6ToobE1hvCms1og__25_OUT1;

  cbd->time_last = cbd->time;
}

void calculateEquations(CBD* cbd) {
  Real delta = cbd->time - cbd->time_last;

  _PIDCLBlock_DzCg6ToobE1hvCms1og__4_IN1 = _PIDCLBlock_x(t);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__4_OUT1 = (-
_PIDCLBlock_DzCg6ToobE1hvCms1og__4_IN1);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__11_OUT1 = 10;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__7_IN2 = _PIDCLBlock_DzCg6ToobE1hvCms1og__11_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__7_IN1 = _PIDCLBlock_DzCg6ToobE1hvCms1og__4_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1 = (_PIDCLBlock_DzCg6ToobE1hvCms1og__7_IN1 +
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_IN2);
  _PIDCLBlock_K_p_OUT1 = 16;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__29_IN2 = _PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__29_IN1 = _PIDCLBlock_K_p_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__29_OUT1 = (_PIDCLBlock_DzCg6ToobE1hvCms1og__29_IN1
* _PIDCLBlock_DzCg6ToobE1hvCms1og__29_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_OUT1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_IN1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delta_t_OUT1 = delta;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delta_t_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_IN1 *
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_OUT1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_IN1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_multDelta_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_IN1 +
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_IN2);
  _PIDCLBlock_K_i_OUT1 = 1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_IN2 = _PIDCLBlock_K_i_OUT1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_OUT1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_OUT1 = (_PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_IN1 *
_PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__21_IN2 = _PIDCLBlock_dqvIs03bip9LbBkwujtJ_2_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__21_IN1 = _PIDCLBlock_DzCg6ToobE1hvCms1og__29_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__21_OUT1 = (_PIDCLBlock_DzCg6ToobE1hvCms1og__21_IN1
+ _PIDCLBlock_DzCg6ToobE1hvCms1og__21_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_OUT1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_IN1;
```

```
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg2_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg2_OUT1 = (-
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg2_IN1);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg2_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_IN1 +
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_delta_t_OUT1 = delta;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_inv_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_delta_t_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_inv_OUT1 = 1.0/
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_inv_IN1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_inv_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum2_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_IN1 *
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_IN2);
  _PIDCLBlock_K_d_OUT1 = 10;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_IN2 = _PIDCLBlock_K_d_OUT1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_mult_OUT1;
  _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_OUT1 = (_PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_IN1
* _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__25_IN2 = _PIDCLBlock_dqvIs03bip9LbBkwujtJ_76_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__25_IN1 = _PIDCLBlock_DzCg6ToobE1hvCms1og__21_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__25_OUT1 = (_PIDCLBlock_DzCg6ToobE1hvCms1og__25_IN1
+ _PIDCLBlock_DzCg6ToobE1hvCms1og__25_IN2);
  _PIDCLBlock_IC_d_OUT1 = 0;
  _PIDCLBlock_IC_i_OUT1 = 0;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_zero_OUT1 = 0;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_delta_t_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_IN1 = _PIDCLBlock_IC_d_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_IN1 *
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg1_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_multIc_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg1_OUT1 = (-
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg1_IN1);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_IN2 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_neg1_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_OUT1 =
(_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_IN1 +
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_IN2);
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__17_delay_IC =
```

```
_PIDCLBlock_DzCg6ToobE1hvCms1og__17_sum1_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_sumState_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayState_IC = _PIDCLBlock_IC_i_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_IN1 =
_PIDCLBlock_DzCg6ToobE1hvCms1og__7_OUT1;
  _PIDCLBlock_DzCg6ToobE1hvCms1og__13_delayIn_IC =
_PIDCLBlock_DzCg6ToobE1hvCms1og__13_zero_OUT1;
  _PIDCLBlock_u(t) = _PIDCLBlock_DzCg6ToobE1hvCms1og__25_OUT1;

  cbd->time_last = cbd->time;
}

void getContinuousStates(CBD* cbd, double x[], size_t nx) {
}

void setContinuousStates(CBD* cbd, const double x[], size_t nx) {
//  calculateEquations(cbd);
}

void getDerivatives(CBD* cbd, double dx[], size_t nx) {
//  calculateEquations(cbd);
}

void getStateEvents(CBD* cbd, double z[], size_t nz) {
  // No state events found
}

void stateEvent(CBD* cbd) {
  // No state events found
  cbd->nominalsOfContinuousStatesChanged = False;
  cbd->terminateSimulation = False;
  cbd->nextEventTimeDefined = False;
}

Status doStep(CBD* cbd, double t, double tNext) {
  // No state events found
  Boolean timeEvent;


  double h = tNext - t;
  while(cbd->time + h < tNext + 0.01 * h) {

    timeEvent = cbd->nextEventTimeDefined && cbd->time >= cbd->nextEventTime;

    if(timeEvent) {
      stateEvent(cbd);
    }

    if(cbd->terminateSimulation) {
      // Force termination
      return Discard;
    }
    cbd->time += h;
    calculateEquations(cbd);
  }
```

```
  return OK;
}
```

`description.xml` describes the model and the general simulation setup. It shows each variable the model has and lists their property, such as the initial values, how their values change, and if they are input, output or a local variable.

`model.c` contains the actual implementation of the model, how the values of variables are calculated and such. The calculations are separated into initial equations and equations, just like in Modelica. Each variable are assigned their initial value in the initial equation function. For each time step, calculateEquations is called and each variable gets their new value.

As the FMU was originally a CBD, we can still see the connections between blocks in the equations. Let's look at the first calculation of the `model.c`file:

```
PIDCLBlock_DzCg6ToobE1hvCms1og__4_IN1 = _PIDCLBlock_x(t);
```

From this we can assume that there is a connection between one of the blocks and the input of the model, called PIDCLBlock_x(t). We can tell that the PIDCLBlock_x(t) is an input of the model by looking at the `causality` property of PIDCLBlock.x(t) in the XML-file.

It is very easy to take multiple FMUs and use them in a single simulation. As each FMU has inputs and outputs, which we can easily get from the XML file, we can just feed the outputs of a model to the input of another model and simulate them separately with the same settings.

2.3 | Compile Controller C-Code and Co-simulate

Now let us try a co-simulation between the plant and CBD we just created.

After some tweaking with the test script, we can get some graphs to compare with the pure Modelica simulation. Do note that there were some errors in the previous Modelica model and has been fixed. The graphs may therefore be different from what you have previously seen. There is also the fact that K_i has been set to 1, instead of 0.

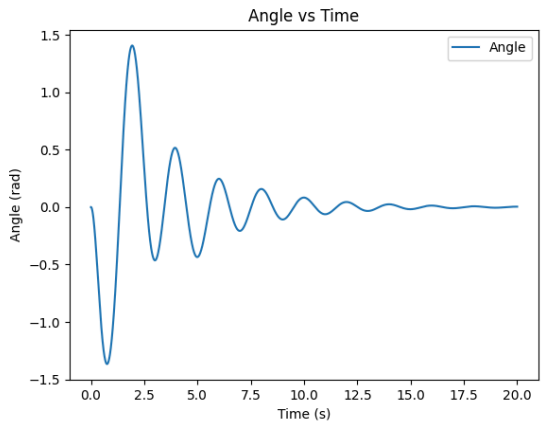

Figure 11: Pendulum angle theta vs time CBD assets/part_2/Angle_vs_Time.png

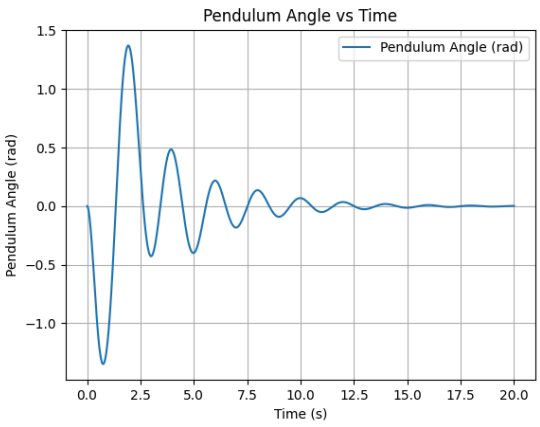

Figure 12: Pendulum angle theta vs time Modelica assets/part_2/ Pendulum_Angle_vs_Time_Modelica.png

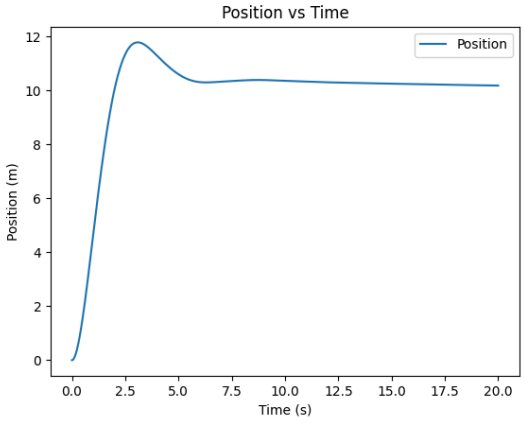

Figure 13: Position X vs time CBD
assets/part_2/Position_vs_Time.png

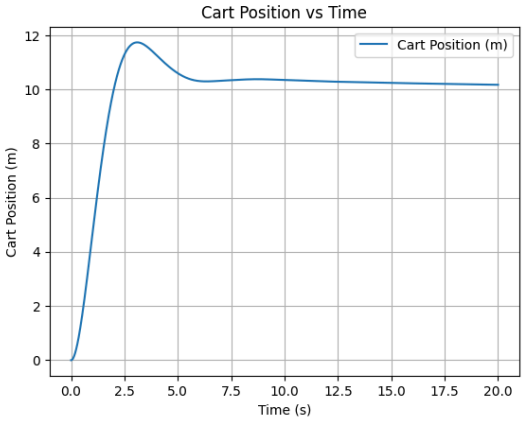

Figure 14: Cart Position X vs time Modelica
assets/part_2/
Cart_Position_vs_Time_Modelica.png

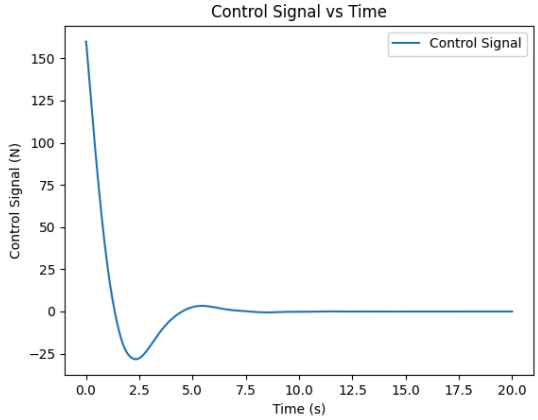

Figure 15: Control signal vs time CBD
assets/part_2/Control_Signal_vs_Time.png

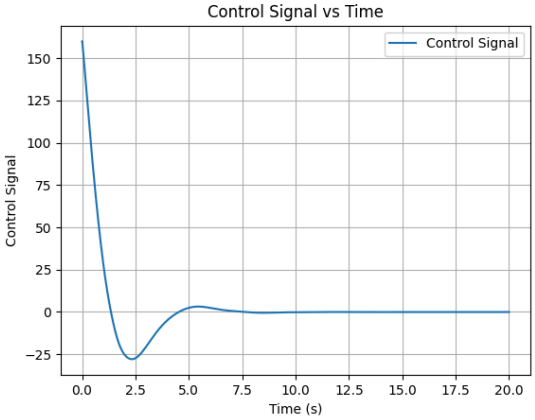

Figure 16: Control signal vs time Modelica
assets/part_2/
Control_Signal_vs_Time_Modelica.png

On the left side we see the CBD models graphs and on the right side we see the modelica graphs.

The similarity between the Modelica and CBD model values is likely due to both models accurately representing the same control loop system dynamics and using consistent mathematical formulations and parameter values.

So the environment are the same, the only difference is “how it is made”. How the equations are computed and how the models will behave are the same.